A Model-based Approach to Developing the Concept of Operations for Potential Mars Sample Return

Sebastian J. I. Herzig, Dianna Velez, Bassem Nairouz, Brian M. Weatherspoon, Raffi P. Tikidjian, Thomas M. Randolph and Brian Muirhead
Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, 91109

Mars Sample Return (MSR) is a proposed multi-agency effort that would return soil and rock samples from the surface of Mars to Earth. Both the complexity of the potential missions, as well as the involvement of multiple geographically distributed organizations, presents a challenge from an information management perspective. In this paper, a Model-based Systems Engineering (MBSE) approach to developing the Concept of Operations of a potential Mars Sample Return effort using the System Modeling Language (SysML) is presented.

Nomenclature

MBSE	=	Model-based Systems Engineering
MSR	=	Mars Sample Return
CONOPS	=	Concept of Operations
SysML	=	Systems Modeling Language
OMG	=	Object Management Group
UML	=	Unified Modeling Language
IMCE	=	Integrated Model-Centric Engineering
SRL	=	Sample Return Lander
ERO	=	Earth Return Orbiter
MAV	=	Mars Ascent Vehicle
JPL	=	Jet Propulsion Laboratory
ESA	=	European Space Agency
NASA	=	National Aeronautics and Space Administration

I. Introduction

Mars Sample Return (MSR) is a proposed effort that would return samples from the surface of Mars to Earth [1]. A combination of robotic systems and a Mars ascent rocket would be used to collect and send Martian rock and soil samples to Earth for detailed chemical and physical analysis. NASA and the European Space Agency (ESA), including the NASA Jet Propulsion Laboratory and several industry contractors, are conducting a study of a joint plan that could accomplish MSR, which is due to the heads of the agencies by 2019. Both the complexity of the proposed campaign itself, as well as the involvement of multiple parties in the design and development process, presents a challenge from an information management perspective. This paper introduces an approach to managing the technical, organizational and programmatic systems engineering information for a potential MSR effort. Specifically, we will introduce our approach to modeling a *concept of operations* (CONOPS), which is meant to act as a basis for functional requirements development.

Current and past NASA JPL missions have faced similar information management challenges. For example, the planned *Asteroid Redirect Robotic Mission* (ARRM) had a systems engineering team that was distributed across multiple NASA centers [2, 3]. Similarly, the planned mission to Jupiter's moon Europa is developed jointly by JPL and the Johns Hopkins Applied Physics Laboratory (APL), leading to similar information management challenges [4, 5]. *Model-based Systems Engineering* (MBSE) [6] has shown to be an effective approach to managing the systems engineering information in both cases. ARRM used the System Modeling Language (SysML) to model a concept of operations, which was also used as a basis for requirements development. The planned mission to Jupiter's moon Europa also uses SysML, but focuses primarily on developing the system architecture. In both cases, the use of MBSE is primarily motivated by the complexity of the overall missions, and the use of SysML by it being a wide-spread, commonly used and actively

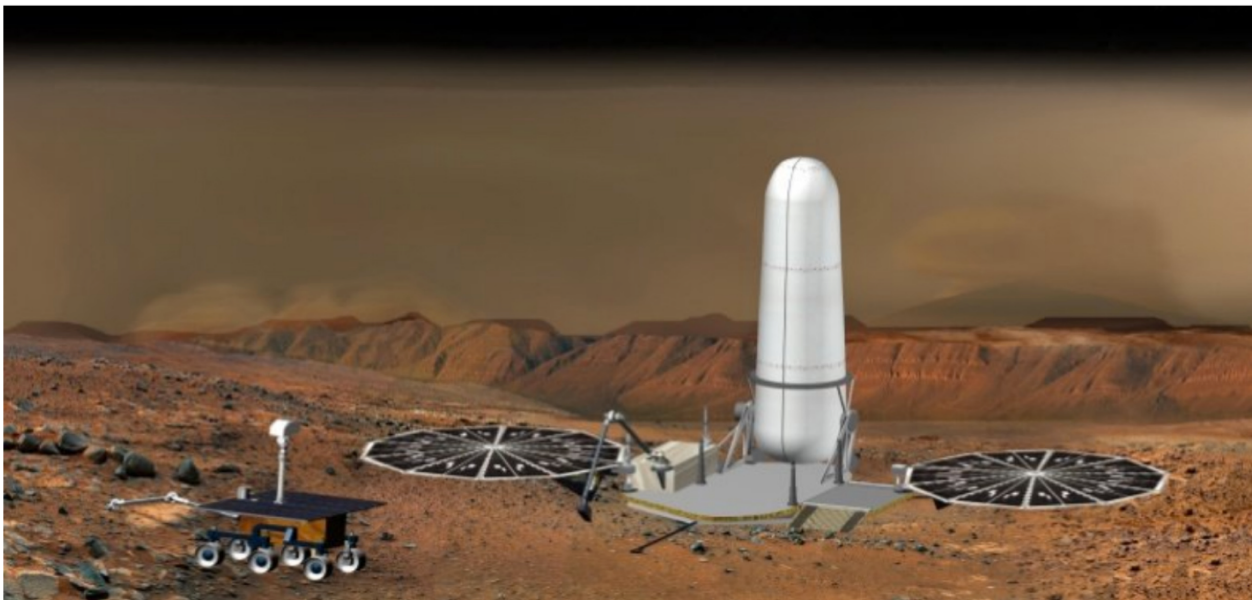

Fig. 1 Artist's concept showing a lander with an attached ascent vehicle contained in a protective covering (right), as well as the envisioned sample fetch rover (left).

maintained system modeling language.

The herein presented approach to developing the concept of operations for a potential MSR effort leverages state of the art methods and tools, the lessons learned and insights gained by past projects, and uses best practices developed at JPL. Specifically, our approach is based on using SysML with a set of syntactic and semantic extensions that were developed internally at JPL as part of the Integrated Model-Centric Engineering (IMCE) effort [7]. The key contribution of this paper is a rigorous, model-based approach and an accompanying set of modeling patterns for modeling a concept of operations using state of the art Model-based Systems Engineering methods and tools.

The paper is organized as follows: section II provides a brief overview of a possible Mars Sample Return mission concept, and gives a definition for what we consider to be a *Concept of Operations*. Other work reporting on the use of model-based methods for developing a concept of operations is then introduced in section III. Thereafter, we introduce our modeling methodology in section IV. Results to date from applying the approach to a potential Mars Sample Return effort are then presented in section V. The paper closes with a brief summary of the main insights gained and provides outlook into future work.

II. Background

A. The Proposed Mars Sample Return Effort

Mars Sample Return (MSR) is a proposed set of mission concepts to collect rock and dust samples on Mars and return them to Earth. Typically, missions studying other planetary bodies carry all scientific instruments along. This not only limits the scope and number of studies that can be performed on the collected samples, but also adds to the complexity of the mission. Returning samples would enable far more exhaustive studies of rock, soil and atmospheric particles in laboratories on Earth. However, bringing these samples back is challenging.

A number of mission concepts for returning samples from the surface of Mars have been studied in the past by a number of agencies. These studies have been motivated by the high expected scientific return on investment [8]. In April 2018, a letter of intent was signed by NASA and ESA that may provide a basis for a joint Mars Sample Return campaign in the late 2020s. One possible mission concept would involve the collection of dozens of samples that are collected and cached by the Mars 2020 rover [9], and would be left on the surface of Mars for possible later retrieval. Two launches are then envisioned to occur in the late 2020s. The first launch would carry a lander with a rover (the *Sample Fetch Rover*), and a rocket (the *Mars Ascent Vehicle* (MAV)) that is capable of bringing a container with samples from the surface into an orbit around Mars. After landing on the surface of Mars, the fetch rover would retrieve the

cached samples and deliver them to the MAV. The MAV would then launch and deliver the sample container into an orbit. The second launch would deliver an orbiter to Mars that rendezvous with the orbiting sample container and delivers it back to Earth. There, the samples would be retrieved and analyzed in specially designed laboratories [10]. See Figure 1 for an artist's concept showing the lander, fetch rover, and ascent vehicle.

B. Concept of Operations (CONOPS)

The *Concept of Operations* (ConOps or CONOPS) describes how a system (or spaceflight mission) will be developed, initiated, operated, and retired during the life-cycle phases to meet stakeholder expectations. Its primary purpose is to describe the system characteristics from an operational perspective and to facilitate an understanding of the system goals. It stimulates the development of the (functional) requirements and informs the architecture of the system. It serves as the basis for subsequent definition documents and provides the foundation for the long-range operational planning activities [11].

Based on a review of the related literature [2, 11, 12], and applicable standards such as IEEE 1362-1998 [13] we define the CONOPS of a spaceflight mission to be composed of the following elements:

- A statement of the **goals and objectives** of the system and/or mission
- **Constraints, policies and strategies** (e.g., communication strategies) affecting the system and/or mission
- A clear statement of the **organizations, participants and stakeholders** involved, as well as the **delegation of responsibilities and authority**
- A series of **operational scenarios** (or *operational concepts*) that describe envisioned processes (flows of events) for initiating, developing, operating and retiring the system

A key element, and often primary focus of a concept of operations are the operational scenarios. Operational scenarios describe envisioned, or governing operational behavior that facilitate reaching goals and objectives of the mission. We define a scenario as an organized set of activities and events together with the associated conditions that govern their progress [11]. A scenario is a step-by-step description of a series of actions and events. These actions and events can occur concurrently and sequentially. Scenarios are an account or synopsys of a projected course of events or actions [13]. Here, we say that actions are single (possibly planned) occurrences of a process, and events are a "thing" that happens, particularly one of importance [11]. Scenarios can be pertinent to certain mission phases or activities, or can be triggered by events or the presence of certain conditions, or can be subject to negotiated intervals of time, resource limits, or other constraints [11]. The concept of an *operations timeline*, which describes the actions and other sequence-related elements necessary to achieve the mission objectives in each phase, is therefore closely related to the concept of operational scenarios.

Part of a CONOPS are also statements about the organizations and stakeholders involved, as well as their responsibilities and authority scopes. This may include a definition of the products and components being developed, and the roles assigned to personnel that are responsible for preparing, concurring or approving components, products, processes and other assigned elements.

III. Related Work

In the related literature on systems engineering research, only few examples can be found in which model-based methods are used to capture a concept of operations. Two related efforts are the design and development of the Alignment and Phasing System (APS) of the Thirty Meter Telescope (TMT) [12], and the Asteroid Robotic Redirect Mission (ARRM) [2, 3].

The Thirty Meter Telescope project has adopted the Model-based Systems Engineering paradigm for a significant part of the system design and development activities. In particular, the APS systems engineering team is developing a system model containing requirements, a logical and physical architecture, software behavior, and other specification elements. The development of this model is driven by operational scenarios, which strictly adhere to the semantics defined in the fUML standard [14] to facilitate executability. Operational scenarios are used to both drive the system design and to verify requirements [15]. The former is done by using operational scenarios as a basis for soliciting the required behavior of the system and identifying and logically grouping functionality into components. Verification is performed by executing operational scenarios, which in turn execute system behavior. Results of simulating the system behavior are then compared to requirements, to which value properties and constraints are attached [12]. These

represent both the value to compare against (e.g., $t_{req} = 180s$) and the condition to be fulfilled (e.g., $t_{sim} \leq t_{req}$). This allows for the analysis of the expected performance (timing, power usage, etc.) of the system as specified and allows for the identification of design errors (deadlocks, unreachable states, race conditions, etc.) early in the life-cycle and often. A principle of the TMT / APS MBSE effort is to use only the vocabulary native to the SysML language. While the system model contains some aspects of authority delegation, the lack of sufficiently precise vocabulary requires human interpretation of some programmatic and technical content. The captured delegation of responsibility is also ambiguous and requires interpretation of the package structure and modeling patterns used in the model. Goals and objectives of APS or TMT, as well as strategies and policies are not explicitly modeled. The TMT systems model is available under an open source license *.

The Model-Based Systems Engineering paradigm has also been successfully used for the conceptual and preliminary design development of the Asteroid Redirect Robotic Mission (ARRM) [2]. Development of a system model in SysML started in early phases of design and initially focused on capturing the Concept of Operations. Focus was primarily placed on developing a set of operational scenarios. Later, the use of the model was extended to capturing the system architecture, a functional decomposition, managing requirements, and tracing requirements to system functions, and as a basis for linking requirements to verification and validation activities. To precisely capture elements of the concept of operations, ARRM utilized both an early version of the SysML profile [16] embedding of the JPL IMCE ontologies, as well as project-specific extensions of this profile. This facilitated the generation of documents and other artifacts from the model. Similar to the proposed Mars Sample Return mission, ARRM faced the challenge of developing a system model collaboratively with geographically distributed partners, in this case with other NASA centers.

IV. Modeling Methodology

The MSR MBSE effort is focused on CONOPS development to provide an architectural framework for initial requirement development, provide early requirements validation by linking requirements to functions and activities, and ultimately provide for the constraints of the function and activity dictionaries used in mission planning and sequence generation.

In the following, we introduce our modeling methodology. First, we introduce the general principles followed in our model development approach. Thereafter, we introduce a number of modeling patterns for capturing the various elements of a concept of operations.

A. Model Development Approach

Proposed is an approach that uses SysML [16] as a modeling language for capturing a concept of operations. This is similar to the approaches taken by ARRM [2] and TMT [12]. We build our approach upon the assumption of having to extend the language vocabulary provided by the SysML language in order to unambiguously capture the various elements of a concept of operations. For this purpose, we use the language extension mechanism native to SysML and UML (the language that SysML is built upon), which foresees the extension of language concepts through the use of *stereotypes* organized in *profiles*. These stereotypes extend existing UML *meta-classes* (such as `Class`, `Actor`, `Activity`, etc.) by allowing named extensions of these concepts to be introduced. For example, the SysML language uses this mechanism to introduce the concept of a *Block*, which is an extension of the UML meta-class `Class`. Note that these extensions are purely syntactic in nature. By using this mechanism we can effectively build a *domain-specific language* that is hosted in a UML-based parent language - in this case, SysML - by embedding concepts specific to a particular domain (here: systems engineering, and concept of operations development) in the host language.

Similar to ARRM, we use the profile embedding of the ontologies developed by the Integrated Model-Centric Engineering (IMCE) effort as a foundation [7]. Following the recommendations of IMCE, we extend the provided vocabulary through specialization of existing stereotypes to account for project-specific needs. This not only extends the available vocabulary of SysML in a syntactic fashion, making it easier for human consumption and to parse by a computer program, but, through a defined mapping to an ontology with the semantics of the description logic $\mathcal{SHOIN}(\mathcal{D})$ [17] also provides a basis for performing logical reasoning (note that this particular flavor of a description logic is a decidable subset of first-order logic, and corresponds to the decidable subset used by OWL-DL [18]). Given appropriate tools, this allows for the computerized detection of logical fallacies in the modeled information which may otherwise be very hard to detect.

*`https://github.com/Open-MBEE/TMT-SysML-Model`

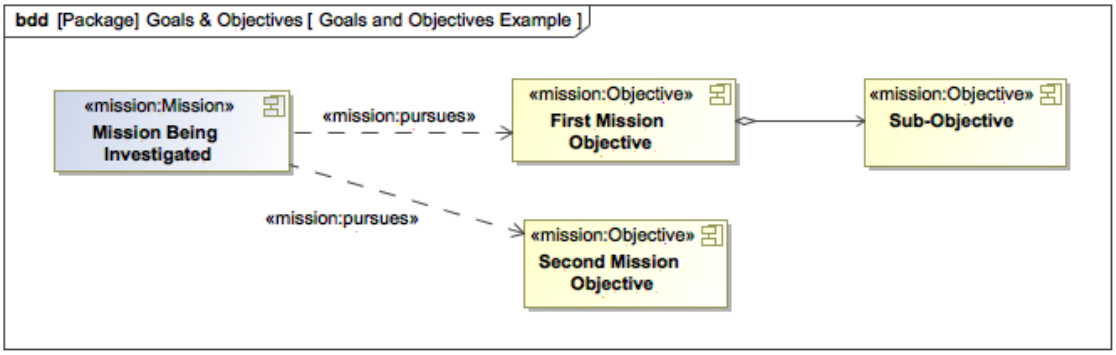

Fig. 2 Illustration of the pattern for modeling the objectives and goals pursued by a particular spaceflight mission.

B. Modeling Patterns

In the following, the proposed modeling patterns for modeling the various elements of a concept of operations (see section II.B) are introduced. For each modeling pattern, we first introduce *what* information we intend to capture. Then, we describe *how* we capture this information using UML / SysML as a host language. Each of the modeling patterns is illustrated using a qualitative example.

Note that, in the following, we use the convention *prefix:Concept* in some instances. This is used to provide the context or namespace in which the particular concept *Concept* is defined. This is done for purposes of disambiguation. For instance, *Activity* is a concept that may be found in a variety of contexts: it may refer to a meta-class in the UML meta-model, or an concept in the realm of a concept of operations. To disambiguate, we refer to these as *uml:Activity* or *conops:Activity*.

1. Mission Goals & Objectives

An important part of a concept of operations is the statement of the goals and objectives of the mission or system being developed. For this pattern, we rely solely on concepts and relations from the IMCE ontologies.

Figure 2 illustrates an example instance of the suggested pattern. Defined is a *Mission* (here: in blue), along with a number of *Objectives* (in yellow). The choice of colors is arbitrary, and only used as a visual aid to distinguish between the different kinds of concepts used. The *Mission* is related to *Objectives* through the *pursues* relationship. The intended meaning is that a Mission *m* pursues an Objective *o* if and only if the successful execution of *m* results in at least partial achievement of *o*. According to the ontological definition, a *Mission* can pursue more than one *Objective*. As illustrated in Figure 2, an *Objective* can also aggregate other *Objectives*.

In the SysML profile provided by IMCE, the concepts *Mission* and *Objective* are embedded as stereotypes extending from the `Component` meta-class in the UML meta-model, and are specializations of the SysML stereotype `Block`. The reason why `Component` is used, and not `Class` (which is the meta-class that the SysML stereotype `Block` natively extend from) is twofold: the semantics of a UML Component more closely match our interpretation of a component (be it physical or logical) being an encapsulated object. Secondly, unlike a UML Class, a UML Component can own a larger variety of elements (including UML Packages and UML Dependencies), enabling greater flexibility when structuring the model and when wanting to identify ownership through namespace membership. *pursues* is embedded as a stereotype that extends meta-class `Dependency`. Aggregate relationships between *Aggregated Elements*, such as are all *Objectives*, are embedded as aggregate associations (i.e., *"white diamond"* relationships). Note that it is assumed that any aggregate association without a stereotype attached is interpreted as an *aggregates* relationship.

2. Operational Scenarios

As mentioned in section II.B, a major part of a typical concept of operations is a set of operational scenarios. Operational scenarios - or operational concepts - were introduced as flows of events that describe envisioned processes for initiating, developing, operating and retiring the system. We propose the use of SysML Activities and SysML Activity Diagrams for this purpose. This is similar to how use cases and scenarios are captured and interpreted in typical applications of UML for software engineering [19], and also similar to the approaches taken by ARRM [2] and

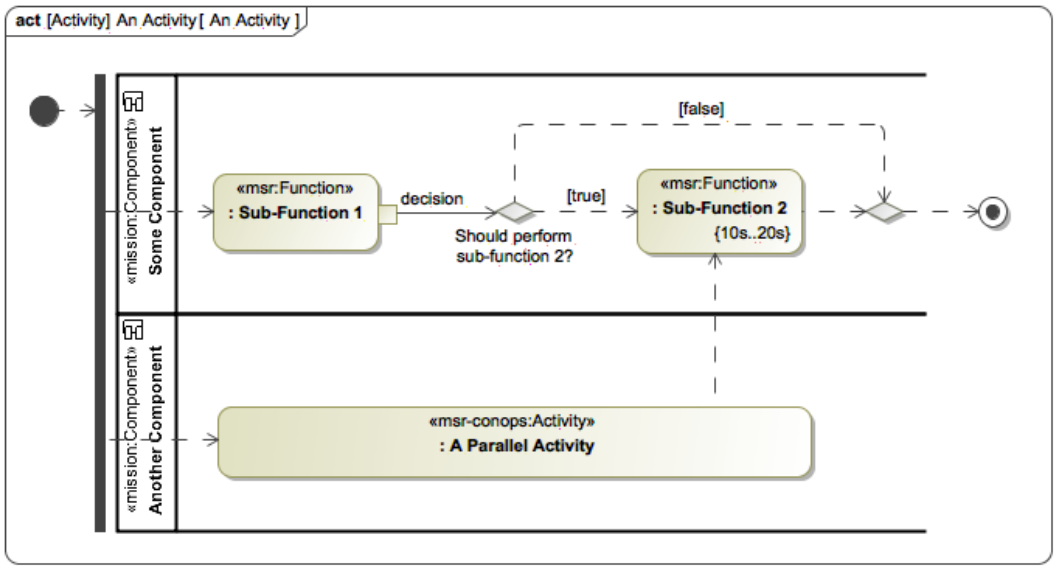

Fig. 3 Example illustration of how a SysML Activity Diagram is used for modeling an operational scenario.

TMT [12].

For modeling operational scenarios of spaceflight missions, we introduce the following new concepts: *Mission Phase*, *Sub-Phase*, *Activity*, *Function*. We declare all concepts subclasses of *mission:Function* as defined in the IMCE *Mission* ontology. We choose to do this because a *Function* is generally defined as an operation or activity performed by a *Component* in the context of executing a *Mission*. Note that a *Mission* itself is defined as a *Performing Element*. This makes it natural to associate mission phases, activities and functions with a mission or one of the components deployed by the mission, and allows for a clear definition of which elements perform what function. The introduction of these project-specific extensions enables a clear differentiation between what is considered a *Mission Phase*, *Sub-Phase*, *Activity* or elemental *Function* performed by a component.

Figure 3 illustrates an example operational scenario. We embed *Mission Phases*, *Sub-Phases*, *Activities*, and *Functions* as stereotypes that extend the `Activity` meta-class. This is different from *mission:Function* which, in the

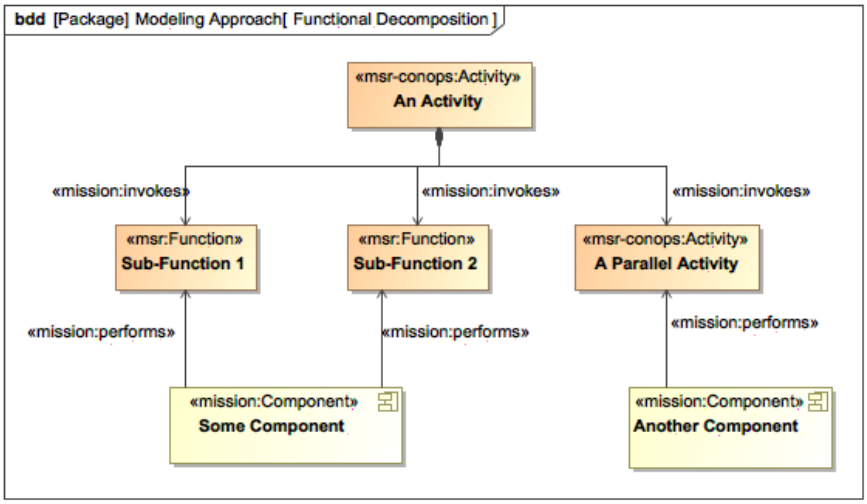

Fig. 4 Example illustration of how composite associations are used to model specific invocations of functions. These invocations are intended to correspond to the CallBehaviorActions in Figure 3. The formal link between the association ends and actions is achieved through SysML *adjunct properties* (not shown explicitly on diagram).

profile provided by IMCE, is embedded as a UML Component. In the scenario, which is defined in the context of an *Activity*, *invocations* of functions (i.e., actions) are modeled using `CallBehaviorActions`. Control flow, guard conditions, merges, forks, and other typical UML Activity diagram elements remain unchanged in their interpretation from what is specified in the fUML standard [14]. These allow timing constraints, and the order of events to be well-defined. UML Duration Constraints (see *Sub-Function 2*: $\{10s..20s\}$) are intended to be used to capture the expected length of time a particular invocation of a function takes to complete. UML Pre- and Post-Condition constraints are used to capture entry and exit conditions. These are specified for each *Mission Phase*, *Sub-Phase*, *Activity* and *Function*, and are assumed to apply to every invocation.

Notice in Figure 3 the use of UML Swimlanes to indicate which *Components perform* a particular *Function*. In UML, swimlanes are purely visual constructs. However, in SysML the concept of an *Allocated Activity Partition* assigns a relationship *allocate* between the element that the swimlane represents, and the UML CallAction (or UML Behavior associate with the UML CallAction). In our proposed approach, we impose a similar interpretation, where the element represented by the swimlane *performs* the particular invoked *Function*. This relationship is illustrated in Figure 4. Also note the use of composite associations (with the appropriate stereotype) to capture the *invokes* relationship between different kinds of *mission:Functions*. This is done such that an invocation can be directly referred to (e.g., in order to point at the invocation using another relationship). To the avid UML modeler, this may not seem intuitive: a `CallBehaviorAction` is owned by a UML Activity, and is not a kind of UML Property. It is also not a UML Association End, and can, therefore, not be directly referenced by the composite association. So how can a UML Association End (e.g., a UML Property owned by the UML Activity) be related to the CallBehaviorAction invoking the particular UML Behavior? The trick is making use of the *adjunct property* concept defined in the SysML language [16]. This allows one to cleanly capture that the UML Association End that the composite association is pointing to is intended to represent a particular UML CallBehaviorAction.

The envisioned behavior captured in an operational scenario is not limited to a single, nominal flow of events. Decision nodes, and other control flow techniques can be used for modeling conditions for off-nominal behavior. This is also shown in Figure 3, where *Sub-Function 2* is only executed if the result of invoking *Sub-Function 1* is the decision to proceed with executing *Sub-Function 2*.

3. Assignment of Responsibilities and Authority Delegation

Operational scenarios are primarily used to model the envisioned flow of events when operating a spacecraft, and to allocate functions (or activities, or sub-phases) to components. However, equally important when defining a concept of operations is defining operational and programmatic content related to development of the system or mission. This includes a statement of the organizations involved, roles, personnel, products, and delegation of authority and responsibility. This is especially important in the context of a possible multi-agency mission such as the proposed Mars Sample Return, since a clear separation of concerns, and clear statement of responsibilities and authority is required to appropriately manage the complexity of the effort. Our approach for modeling these elements is discussed in the following.

The IMCE *project* ontology defines the concepts and relations needed for capturing what was referred to in section II.B as *"a clear statement of the organizations, participants and stakeholders involved, as well as the delegation of responsibilities and authority"*. Figure 5 shows an example of the kinds of elements relevant for our context, and the kinds of relationships that can be established between these. *Organizations* are defined to have *responsibility* for various *Authorities* such as *Projects*, *Work Packages*, and *Programs*. *Persons* can belong to zero or more *Organizations*, and can have zero or more *Roles*. *Products* are *prepared*, or *approved* by *Roles*. *Roles* can also *delegate to* other *Roles*. *Authorities* can *authorize* other *Authorities*, and *Work Packages produce Products*. Note that this provides only an excerpt of the vocabulary provided by the project ontology. More examples are provided later in the paper. As before, this vocabulary allows for a precise, rigorous and unambiguous definition of authority delegation and delegation of responsibilities. Note that different representations and views on this, or a subset of the information may be necessary for different purposes. For instance, such views may include a matrix or tree structure showing the assignment of roles to persons. Note how it is *Roles* that have assignments, and not *Persons*. This is practical, since *Persons* can have different roles during a project life cycle, and the roles assigned to *Persons* can switch during a project's life cycle frequently. In other words, the *Roles* in a project are more likely to be stable than the Persons.

In our approach, the concept of a *Role* is embedded as a stereotype extending from the UML meta-class `Actor`. The concepts of *Product*, *Organization* and *Person* are all embedded as stereotypes extending from the UML meta-class `Component`, and specializing the SysML stereotype `Block`. The concept of an *Authority* (e.g., *Work Package*, *Project*)

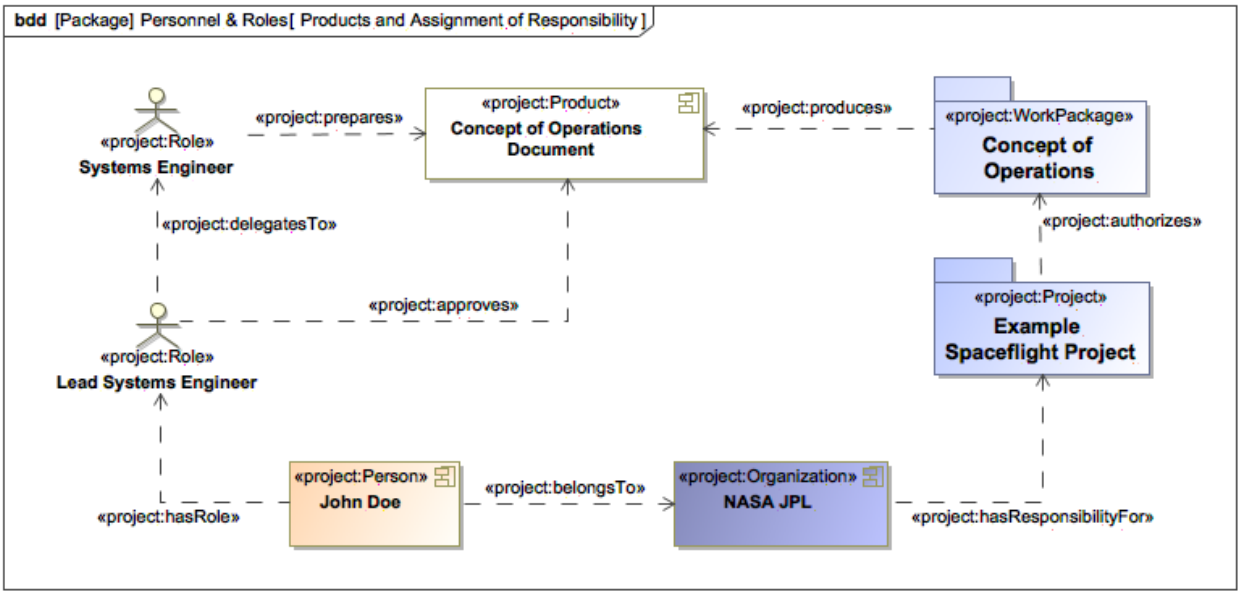

Fig. 5 Illustration of how roles, personnel, organizations, products, and various authorities (here: work packages and projects) are proposed to be modeled and related to one another. Generally, roles have assignments (such as the preparation or approval of a product), and persons have roles. Persons also belong to organizations, which have responsibilities over certain authorities. Work packages produce products.

is embedded as a stereotype that extends the UML meta-class `Package`. This allows for a natural grouping of elements by authority. We assign the meaning of any element being contained in a particular *Authority* (i.e., in an appropriately stereotyped UML Package) to be that the contained element is *authorized* by the *Authority*. For instance, if the element *Concept of Operations Document* in Figure 5 is contained in the package *Example Spaceflight Project*, then this is interpreted as the *Example Spaceflight Project* authorizing this particular product.

4. Constraints, Policies and Strategies

As identified in section II.B, constraints, policies and strategies (e.g., communication strategies) affecting the system and/or mission should also be included in a concept of operations. In our modeling methodology, most of these elements are strongly linked to other modeling patterns. For instance, constraints are included as part of operational scenarios, either as timing constraints on individual invocations of activities, or as entry- & exit-conditions. Other policies, strategies and constraints may need to be modeled as interfaces between elements.

Figure 6 shows one related element commonly found in a concept of operations of spaceflight mission: the fact that a particular environment affects a particular mission concept. For instance, a spacecraft intended to be traveling to a

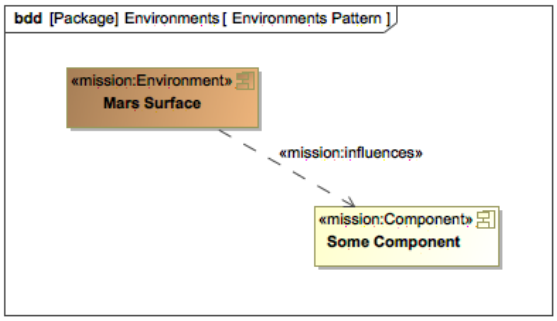

Fig. 6 Illustration of the modeling pattern relating environments to components.

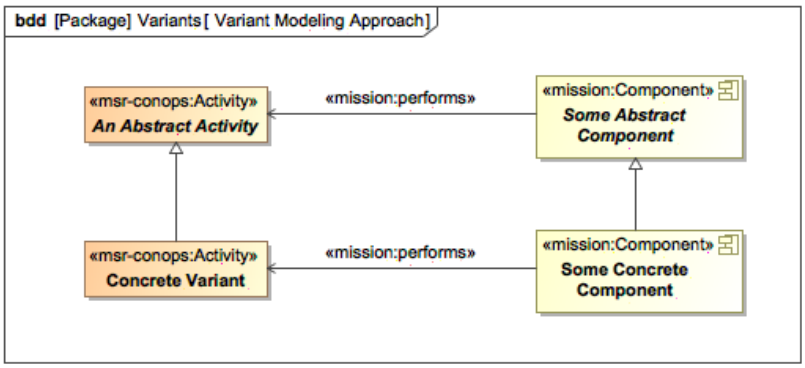

Fig. 7 Illustration of how open trades are proposed to be captured in the context of a concept of operations.

planetary body such as Mars will be exposed to a deep space environment. Similarly, the environment on the Mars surface will influence a rover. Components may also *induce* environments. For instance, a rocket may induce a dust environment on Mars during launch. We use this to more precisely capture the conditions that different components participating in an operational scenarios are exposed to.

The necessary vocabulary for defining the relationship between environments and components is defined in the IMCE *mission* ontology. The concept of an *Environment* is embedded as a stereotype that extends from the UML meta-class `Component`. *influences* and *induces* are embedded as stereotypes extending from the meta-class `Dependency`.

5. Options & Variants

Trades are an important part of design, particularly in early life-cycle phases. Unfortunately, the UML and SysML languages do not have a sufficiently comprehensive mechanism for capturing design alternatives [20]. However, a mechanism is needed for our purposes, since design choices can have a large impact on how a particular component is utilized in achieving a particular mission objective. Here, we briefly introduce our approach for capturing such variation when modeling a concept of operations.

To motivate the importance of variation and capturing options in a concept of operations, consider the following trade: say that a rocket is to be used for bringing Mars rock and dust samples into an orbit around Mars. At the level of abstraction of knowing no more about a design other than there being a "rocket", a scenario describing its launch and ascent may focus solely on abstract activities such as "ignite engine", "ascend", "orbit injection", and "deploy payload". However, say both a single stage hybrid rocket and a two-stage rocket with solid propellant are considered in the design trade space. While the abstract scenario is still considered valid, a number of details for each particular variant will be different, including the sequence of events. For instance, the single stage hybrid rocket may require a different initialization procedure - perhaps even a procedure to heat the rocket to its operational temperature. Similarly, the ascend of each rocket would be fundamentally different, in that an operational scenario involving the launch of the two-stage rocket would include a stage separation activity followed by an ignition of the upper stage. Similarly, consider the choice of propulsion system used to transit from Earth to Mars: the activities involved when using electric propulsion are fundamentally different from those needed when using a more conventional approach such as chemical propulsion. The goal of this modeling pattern is to capture such variation and to formally relate abstract scenarios to more specific variant scenarios and variant system configurations.

We postulate that it is always possible to identify a scenario that represents some*common denominator*. Given the validity of this hypothesis, we can argue that everything that is true for this common denominator, must also be true for the more specific variant. This argument motivates the use of *inheritance* and *generalization / specialization* techniques for modeling variation points (i.e., the "common denominator") and variants (i.e., the more specific elements). This is illustrated in Figure 7. In addition to inheritance of members, the UML language supplies a number of useful techniques (such as redefinition) for carefully establishing the relationships between the superclass (here interpreted as the variation point) and its immediate subclasses (here interpreted as variants).

V. Results to Date

The approach and modeling patterns described in section IV have been applied to the modeling of the concept of operations of a potential multi-agency Mars Sample Return effort. Initial results and excerpts of the model content are presented and discussed in the following. Note that, to date, the focus has been on modeling operational scenarios.

A. Operational Scenarios

To date, 78 operational scenarios have been modeled. These scenarios span all envisioned mission phases of a potential Mars Sample Return, but do not yet provide a sufficiently complete picture of the overall mission. Particularly off-nominal cases have not yet been modeled. The 78 operational scenarios cover a total of 504 mission phases, sub-phases, activities, and functions. Due to the large number and scope of the operational scenarios, a complete overview cannot be given in this paper, and only select aspects are presented.

Figure 8 illustrates what may be considered the primary operational scenario: the envisioned, possible sequence of mission phases for a possible multi-launch Mars Sample Return campaign. The flow of events reflects the mission concept described in section II.A. This diagram has been heavily stylized for presentation purposes, but the underlying activity still strictly adheres to SysML and UML semantics. In the diagram, each `CallBehaviorAction` is presented by an icon, where the icon is associated with the mission phase being invoked. Notice the use of swimlanes in the diagram to capture the environments that the various mission phases are envisioned to take place in. While not a formal relationship, this is intended to inform what environments the components participating in the mission phases are influenced by. Also note how the diagram illustrates how the activities performed by three distinct elements of the mission - the Sample Return Lander (SRL), the Earth Return Orbiter (ERO), and the Sample Processing Facility (SPF) - fuse. One of these points is the *Rendezvous* phase, which represents the critical mission phase where SRL and ERO interact. In this phase, the orbiting sample closes in on the Earth Return Orbiter, and is eventually captured by the Capture / Containment and Return System (CCRS). This is a process that is closely monitored by ground facilities. Figure 9 illustrates a possible scenario for this capture sequence. Notice how swimlanes are used to define which activities are performed by which particular component. Also note that there is no commitment to *how* these activities are performed - i.e., they are kept *functional* and not tied to any particular physical implementation. For instance, *Detect OS Entering CCRS* could be performed by a laser curtain, by a camera, or some other sensor.

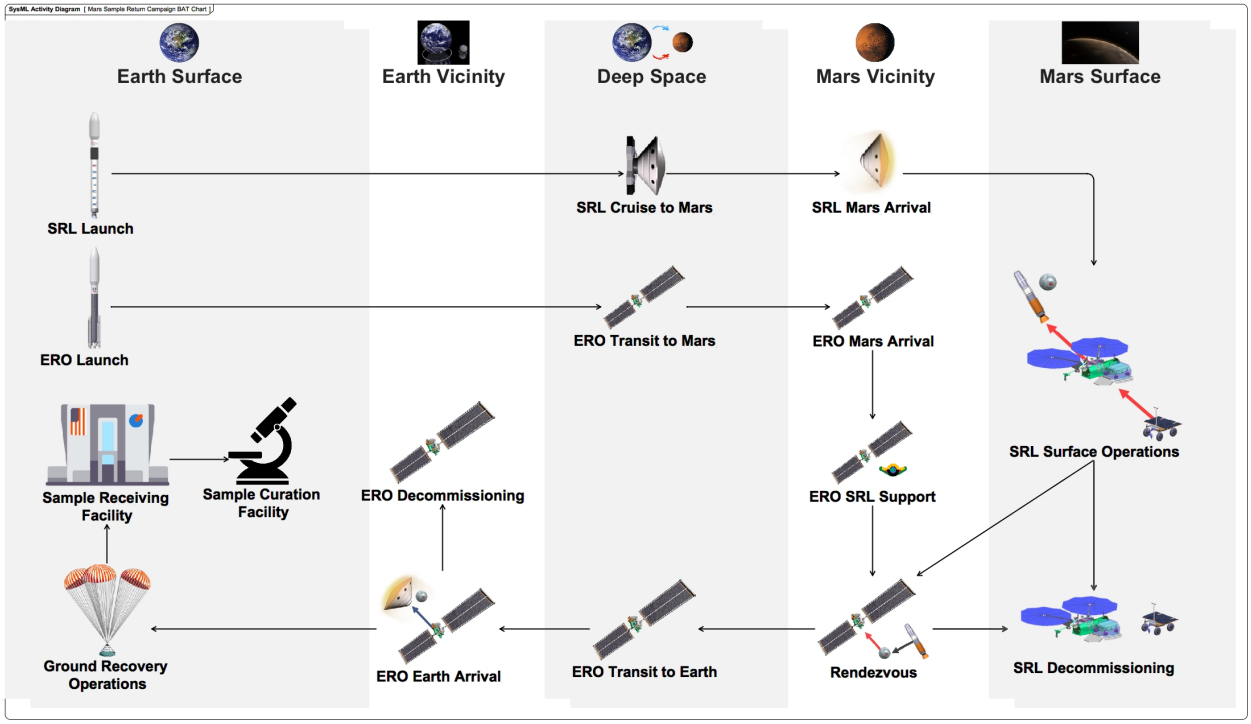

Fig. 8 One possible operational scenario of a potential Mars Sample Return mission.

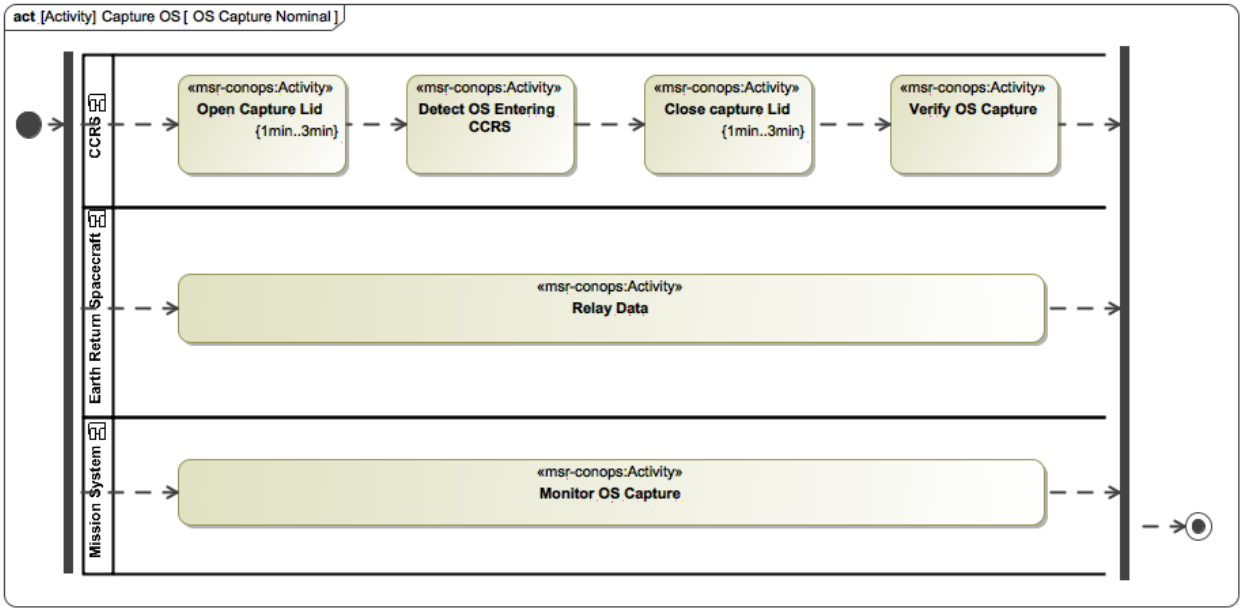

Fig. 9 Operational scenario showing the envisioned process of capturing the orbiting sample, from the perspective of the Earth Return Orbiter project *(not to be considered the actual, finalized specification or approved process, but a fictitious flow of events)*.

Nine of the 78 operational scenarios that have been modeled to date include variation. These variant scenarios show the sequence of events for major possible system trades, including the choice of propulsion system used by the Earth Return Orbiter, whether or not a parachute should be included in a potential Earth Entry Vehicle, the choice of propulsion system for the Mars Ascent Vehicle, and whether a Skycrane (similar to Mars Science Laboratory (MSL)) or landed platform should be utilized to deploy the lander on Mars. These variation points and variants are summarized in Table 1.

As per the approach detailed in section IV.B.5, variants for scenarios are modeled by creating an abstract activity describing the common denominator in terms of envisioned behavior, and one or more specializations of the scenario. This has proven to be very useful. However, care must be exerted when modeling variants: if the variation point is introduced too early, there is a danger of duplicating scenarios. The goal should be to minimize the number of scenarios, and introduce the variation as late as possible.

B. Delegation of Authority & Assignment of Responsibilities

Some aspects of delegation of authority and assignment of responsibilities have been modeled as well. This has become increasingly important as collaboration between agencies and centers has increased. Primary motivation for this has been to clearly identify and capture responsibilities of the various organizations involved in refining the mission concept, and to segregate information as much as possible.

Some non-trivial relationships have been captured in this way. Consider Figure 10, where the delegation of responsibility and authority for the potential Fetch Rover is illustrated. Modeled is the following information: NASA JPL is responsible for the Sample Return Lander Project as a whole, which defines the *Fetch Rover* component. At the project-level, this may include an allocation of resources (such as mass, cost) to the particular component (not shown). Specifying the Fetch Rover at the project-level is the assignment of the *Systems Engineer* shown on the diagram. *John Doe* has this role, and belongs to *NASA JPL*. Interesting to note on the diagram is that the Fetch Rover itself is *realized* by another Work Package that is the responsibility of ESA. As illustrated, this effectively shows the "handing off" of a specification by one organization to another, who responds to the specification with a design. Without extending the vocabulary of SysML, modeling these statements precisely is not possible without imposing a subjective interpretation of the model.

To date, the major campaign, project, spacecraft and subsystem elements have been modeled (as authorities, and components). Most involved organizations have also been captured, and a preliminary version of the associations between

Table 1 Variant scenarios modeled to date.

Variation Point	Variants
Earth EDL	Earth EDL with Parachute
	Earth EDL without Parachute
ERO Cruise to Earth	ERO Cruise to Earth with Chemical Propulsion
	ERO Cruise to Earth with Electric Propulsion
ERO Mars Departure	ERO Mars Departure with Chemical Propulsion
	ERO Mars Departure with Electric Propulsion
ERO Mars Orbit Insertion	ERO Mars Orbit Insertion with Chemical Propulsion
	ERO Mars Orbit Insertion with Electric Propulsion
ERO Transit to Mars	ERO Transit to Mars with Chemical Propulsion
	ERO Transit to Mars with Electric Propulsion
SRL EDL	SRL EDL with Skycrane
	SRL EDL with Landed Platform
OS Terminal Approach	Ballistic Terminal Approach
	Forced Motion Approach
Transfer to Co-Planar Orbit	Transfer to Co-Planar Orbit with Chemical Propulsion
	Transfer to Co-Planar Orbit with Electric Propulsion
MAV Launch	Single-Stage MAV Launch with Hybrid Propulsion
	Dual-Stage MAV Launch with Solid Rocket

projects, work packages and organizations have been modeled. A software tool was written to synchronize personnel with a *Lightweight Directory Access Protocol* (LDAP) list, which is a commonly used access control mechanism. This allows for most easier management of personnel associated with a project. Only a limited number of roles have been modeled to date.

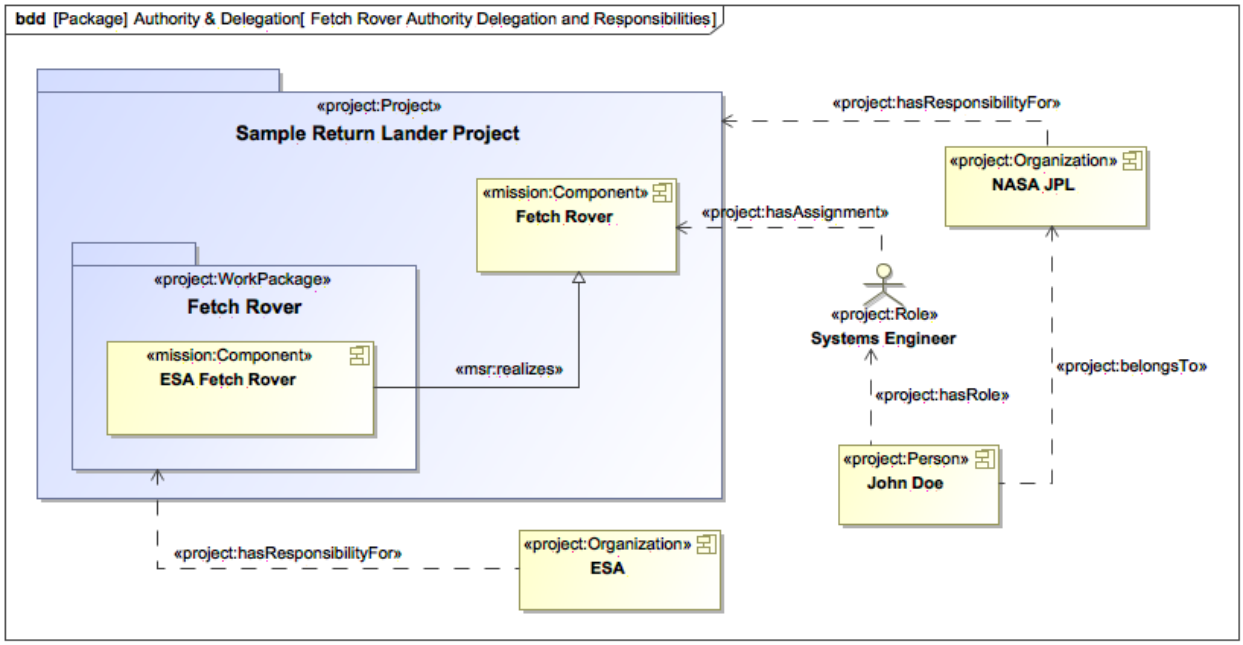

Fig. 10 Assignment of responsibilities and delegation of authority for the case of the proposed Fetch Rover.

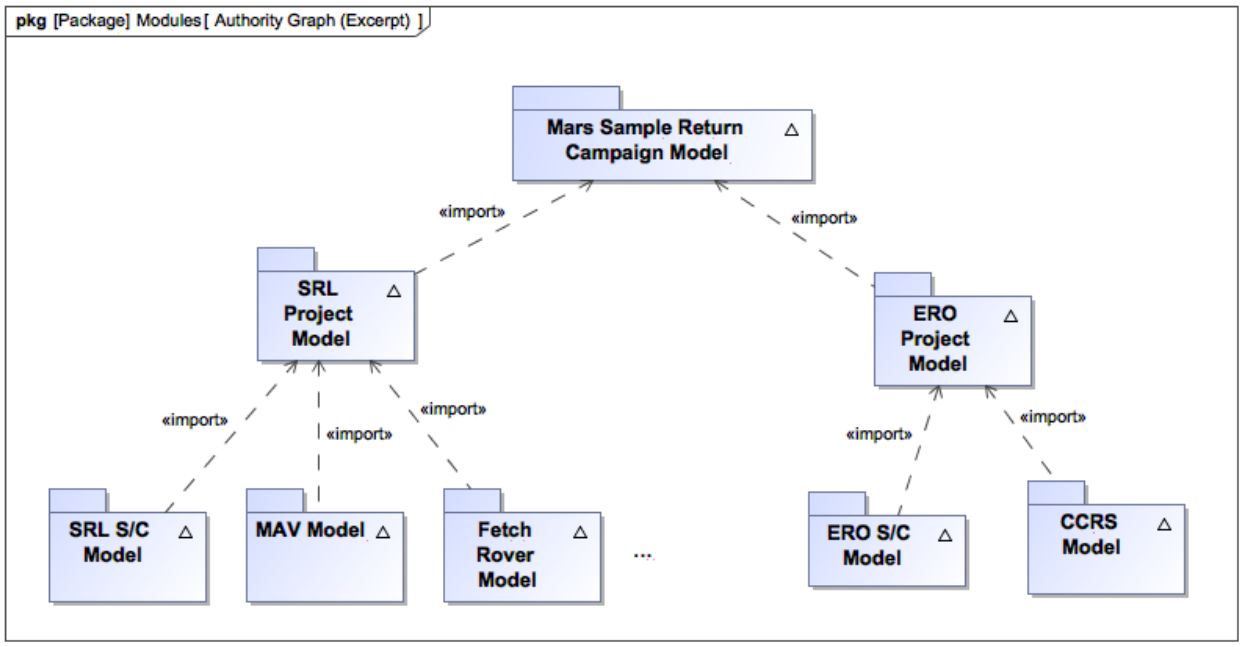

Fig. 11 Excerpt of the model organization for the concept of operations model for a potential Mars Sample Return mission.

C. Model & Package Organization

The overall concept of operations model has been split into 12 separate models, each of which contains information that can be clearly associated with a particular space agency, supplier or NASA center, and is owned by that particular entity. An excerpt of the model hierarchy, and associated links is illustrated in Figure 11. The *Mars Sample Return Campaign Model* is the responsibility of JPL, and includes information controlled by the campaign leadership team. This includes a definition of all mission phases, and associated operational scenarios. The SRL and ERO project models contain project-level information, which includes operational scenarios for various phases associated with SRL and ERO. For instance, an operational scenario detailing the envisioned sequence of events for *Entry Descent and Landing* (EDL) on Mars may be a part of the SRL Project Model. Models lower in the hierarchy describe details pertaining to only the particular flight system or payload system. For instance, the CCRS (Containment / Capture and Return System) Model may include specific scenarios detailing the envisioned steps that lead to the orbiting sample being placed inside a potential Earth Entry Vehicle.

Note that this model hierarchy has a dual purpose: it not only provides a clear separation of concerns by organization, project and subsystem, but it also serves as a basis for segregating model content in such a fashion that proprietary and export controlled information can be shielded off. While this topic is worthy of a paper on its own, the following motivational scenario is provided: assume that, in a NASA-ESA collaboration, all information at the campaign and project level can be shared, but, due to export regulations (e.g., *International Traffic in Arms Regulations* (ITAR)), detailed design information about the Mars Ascent Vehicle cannot. In this case, segregating the detailed design information about the Mars Ascent Vehicle into its own model is beneficial. Similarly, detailed design information about a component developed by ESA may not be shareable with NASA due to export regulations. Information about interfaces between these components, and specifications of how the components shall interact can be shared in a model higher in the hierarchy: e.g., at the project level. This has the positive side effect of managing some of the complexity associated with the overall mission concept by helping to clearly define system boundaries, interfaces, responsibilities, authority and ownership.

VI. Conclusion

In this paper, an approach to modeling a concept of operations is presented, with specific applications to modeling the concept of operations of a potential multi-agency Mars Sample Return effort. The primary purpose of the modeling

effort is the capturing and management of information, and to support functional requirements development. It is intended to act as a basis for requirements management, document generation, and the generation of function and activity dictionaries. The presented approach is based on the use of the System Modeling Language (SysML) along with syntactic extensions provided by JPL's Integrated Model-Centric Engineering (IMCE) effort. We introduced a number of modeling patterns that allow for basic constituents of a concept of operations to be modeled. The paper also shows an application of these patterns to a potential Mars Sample Return, and discusses insights gained from applying it to the development of the concept of operations for this proposed effort.

The use of an extended vocabulary has allowed for capturing the information pertinent to a concept of operations precisely and rigorously. Without the syntactic extensions, we could not have differentiated between, e.g., mission phases and activities, or persons and spacecraft components and products, or clearly identified responsibilities, and the delegation of authority. A consequence of the added precision is an increased set of modeling constructs. This can have both positive and negative side effects: individual pieces of information can be precisely captured in the model, making the model significantly more understandable. For instance, without the vocabulary extensions, capturing that a particular function is performed by a component would have to be modeled using the *Activity allocated to Block* pattern commonly found in SysML reference books [21]. However, since a SysML Activity can also constitute behavior, this is not always unambiguous. What may initially be perceived as a negative consequence of an extended vocabulary is having to have an agreement on the definition of a fairly extensive set of concepts and their relationships. This is challenging, particularly if multiple organizations are involved. However, at least in the case of a highly complex systems engineering effort, where a precise understanding of concepts is key, this can have a high return in investment, since a universal *language* is formed that all stakeholders in a project agree on and use. This facilitates communication, and can reduce ambiguity even in other, informal contexts (i.e., in documents, presentations, etc.).

Finding a model structure early on that segregates information by ownership of the different organizations involved has proven to be very useful thus far. Identifying clear interfaces has positively supported the overall systems engineering effort. Segregating information by organizational ownership has shown to be a good basis for developing an overall concept of operations in a distributed fashion. However, given the early stage of the collaborative effort, further investigation is warranted.

Future work should include further investigating the effects of developing a distributed concept of operations model, where different parts are developed by geographically distributed organizations. Also investigated should be making use of the semantic mapping of the utilized ontologies to OWL for purposes of detecting logical fallacies in the concept of operations (e.g., cylic delegations of responsibility). Furthermore, future work should include how the model can be extended to go beyond just content related to the concept of operations. For instance, investigated should be how system behavior specifications can be tied to operational scenarios for purposes of verification, similar to what was achieved with TMT [12]. Also investigated should be alternative means of data entry and information visualization for stakeholders with varying degrees of expertise in the utilized modeling languages and tools.

Acknowledgments

This task was managed out of the Jet Propulsion Laboratory, a division of the California Institute of Technology, under a contract with the National Aeronautics and Space Administration. The information presented about potential Mars sample return architectures is provided for planning and discussion purposes only. NASA has made no official decision to implement Mars sample return. The authors would like to thank the many people who have provided input and feedback to this effort including, but not limited to, Robert Lock, Patrick J. Guske, Hans-Peter de Koning, Jakob Huesing, Humphrey Price, Austin Nichols, Steven Jenkins, and David Wagner. The work described in this paper leverages MBSE infrastructure components developed as part of the NASA JPL Integrated Model-Centric Engineering (IMCE) initiative, Systems & Software Computer Aided Engineering (SSCAE), the Asteroid Redirect Robotic Mission (ARRM), the planned mission to Jupiter's moon Europa, and other JPL institutional systems engineering modernization efforts. The authors would like to acknowledge the input provided by the members of the Mars Sample Return study team.

References

[1] "Mars Sample Return Informational Page," `https://www.jpl.nasa.gov/missions/mars-sample-return-msr/`, Mar. 2018.

[2] Sindiy, O., Mozafari, T., and Budney, C., "Application of Model-Based Systems Engineering for the Development of the Asteroid Redirect Robotic Mission," *AIAA SPACE 2016*, 2016, p. 5312.

[3] Sindiy, O., Weatherspoon, B., Tikidjian, R., and Mozafari, T., "Extension of MBSE for Project Programmatics Management on the Asteroid Redirect Robotic Mission," *Aerospace Conference, 2017 IEEE*, IEEE, 2017, pp. 1–11.

[4] Dubos, G. F., Coren, D. P., Kerzhner, A., Chung, S. H., and Castet, J.-F., "Modeling of the Flight System Design in the Early Formulation of the Europa Project," *Aerospace Conference, 2016 IEEE*, IEEE, 2016, pp. 1–14.

[5] Dubos, G., Schreiner, S., Wagner, D. A., Jones, G., Kerzhner, A. A., and Kaderka, J., "Architecture Modeling on the Europa Project," *AIAA SPACE 2016*, 2016, p. 5310.

[6] Fisher, J., et al., "Model-based Systems Engineering: a New Paradigm," *INCOSE Insight*, Vol. 1, No. 3, 1998, pp. 3–16.

[7] Bayer, T. J., Cooney, L. A., Delp, C. L., Dutenhoffer, C. A., Gostelow, R. D., Ingham, M. D., Jenkins, J. S., and Smith, B. S., "An Operations Concept for Integrated Model-Centric Engineering at JPL," *Aerospace Conference, 2010 IEEE*, IEEE, 2010, pp. 1–14.

[8] National Research Council, "Vision and Voyages for Planetary Science in the Decade 2013-2022," Tech. rep., National Aeronautics and Space Administration (NASA), 2011. URL `http://solarsystem.nasa.gov/2013decadal/`.

[9] Mustard, J., Adler, M., Allwood, A., Bass, D., Beaty, D., Bell, J., Brinckerhoff, W., Carr, M., Des Marais, D., Brake, B., et al., "Report of the Mars 2020 Science Definition Team," *Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA*, 2013.

[10] Mattingly, R., and May, L., "Mars Sample Return as a Campaign," *Aerospace Conference, 2011 IEEE*, IEEE, 2011, pp. 1–13.

[11] Hirshorn, S. R., Voss, L. D., and Bromley, L. K., "NASA Systems Engineering Handbook," 2017.

[12] Herzig, S. J. I., Karban, R., Trancho, G., Dekens, F. G., Jankevičius, N., and Troy, M., "Analyzing the Operational Behavior of the Alignment and Phasing System of the Thirty Meter Telescope using SysML," *Proceedings of Adaptive Optics for Extremely Large Telescopes (AO4ELT)*, 2017.

[13] "IEEE Guide for Information Technology - System Definition - Concept of Operations (ConOps) Document," *IEEE Std. 1362-1998*, 2007.

[14] "Semantics Of A Foundational Subset For Executable UML Models (FUML), Version 1.1, Beta 1," `http://www.omg.org/spec/FUML/1.1/`, october 2012.

[15] Karban, R., Jankevičius, N., and Elaasar, M., "ESEM: Automated Systems Analysis using Executable SysML Modeling Patterns," *INCOSE International Symposium*, Vol. 26, Wiley Online Library, 2016, pp. 1–24.

[16] "The OMG System Modeling Language Specification," `https://www.omg.org/spec/SysML/`, Mar. 2018.

[17] Baader, F., Calvanese, D., McGuinness, D., Patel-Schneider, P., and Nardi, D., *The Description Logic Handbook: Theory, Implementation and Applications*, Cambridge University Press, 2003.

[18] Horrocks, I., and Patel-Schneider, P. F., "Reducing OWL Entailment to Description Logic Satisfiability," *International Semantic Web Conference*, Springer, 2003, pp. 17–29.

[19] Bruegge, B., and Dutoit, A. H., *Object-Oriented Software Engineering Using UML, Patterns and Java*, Vol. 2004, Prentice Hall, 2004.

[20] Berger, T., Rublack, R., Nair, D., Atlee, J. M., Becker, M., Czarnecki, K., and Wąsowski, A., "A Survey of Variability Modeling in Industrial Practice," *Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems*, ACM, 2013, p. 7.

[21] Friedenthal, S., Moore, A., and Steiner, R., *A Practical Guide to SysML: the Systems Modeling Language*, Morgan Kaufmann, 2014.